Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı, S. 93-101, Ekim 2019
© Telif hakkı EJOSAT'a aittir
Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue, pp. 93-101, October 2019
Copyright © 2019 EJOSAT
Research Article

Yüksek Boyutlu Test Fonksiyonlarında Ağaç Tohum Algoritmasının Performans Analizi

Mehmet Beşkirli[1]

[1] Şırnak Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Şırnak, Türkiye (ORCID: 0000-0002-4842-3817)

(Bu yayın HORA 2019 kongresinde sözlü olarak sunulmuştur.)

(First received 1 August 2019 and in final form 22 October 2019)

(**DOI:** 10.31590/ejosat.636416)

ATIF/REFERENCE: Beşkirli, M. (2019). Yüksek Boyutlu Test Fonksiyonlarında Ağaç Tohum Algoritmasının Performans Analizi. *European Journal of Science and Technology*, (Özel Sayı), 93-101.

Öz

Gelişen teknoloji ile birlikte ortaya çıkan problemlerin klasik yöntemler ile çözümü gün geçtikçe zorlaşmaktadır. Klasik optimizasyon teknikleri, problem boyutunun büyük olması ve çözüm uzayının geniş olması gibi nedenlerden dolayı yetersiz kalmaktadır. Bu nedenle klasik optimizasyon metotları ile gerçek dünya problemlerini çözmek çok zaman almakta ve etkili bir şekilde çözülememektedir. Bu gibi sorunlar insanoğlunu kesin çözüme değil optimizasyon yöntemleri ile bulunabilecek en iyi çözüme doğru yöneltmektedir. Optimizasyon, eldeki kısıtlı imkanları en iyi şekilde kullanarak mümkün olan en iyi sonucu elde etme olarak tanımlanmaktadır. Optimizasyon yöntemi meta-sezgisel algoritmalar kullanılarak gerçekleştirilmektedir. Böylece genel olarak popülasyon tabanlı algoritmalar olarak adlandırılan birçok yeni algoritma önerilmiştir. Popülasyon tabanlı olan ağaç tohum algoritması meta-sezgisel algoritmalardan bir tanesidir ve sürekli optimizasyon problemlerini çözmek için kullanılan ve aynı zamanda yeni çıkan algoritmalardan biridir. Ağaç tohum algoritması ağaçlar ile tohumlar arasındaki bağlantıdan esinlenmiş olup her çevrimde her ağaç için belirli sayıda tohum üretilmektedir. Bu çalışmada ağaç tohum algoritması ile geliştirilmiş ağaç tohum algoritması on farklı test fonksiyonları için kullanılmıştır. Problem boyut 100 olarak alınmış ve algoritmanın popülasyon sayıları sırasıyla 10, 30 ve 50 olarak kabul edilmiştir. Algoritmanın ST parametre değeri orijinal hali olan 0.1 değeri kabul edilmiştir. Algoritmalar aynı koşullarda 30 kez MATLAB programında çalıştırılmıştır. Elde edilen sonuçlar incelendiğinde geliştirilmiş ağaç tohum algoritmasının orijinal ağaç tohum algoritmasına göre daha iyi bir sonuç elde ettiği görülmüştür.

Anahtar Kelimeler: Ağaç Tohum Algoritması, Optimizasyon, Test Fonksiyonları.

Performance Analysis of Tree Seed Algorithm in High Dimensional Test Functions

Abstract

With the developing technology, the problems that arise with classical methods become more and more difficult. Classical optimization techniques are inadequate due to the large size of the problem and the large solution space. Therefore, solving real world problems with classical optimization methods is time consuming and cannot be solved effectively. Such problems lead human beings to the best solution that can be found by optimization methods rather than the exact solution. Optimization is defined as achieving the best possible results by making the most of the limited possibilities available. Optimization method is realized by using meta-heuristic algorithms. Thus, many new algorithms, generally called population-based algorithms, have been proposed. The population-based tree seed algorithm is one of the meta-heuristic algorithms and is one of the new algorithms used to solve continuous optimization problems. The tree seed algorithm is inspired by the connection between the trees and the seeds and a certain number of seeds are produced for each tree per cycle. In this study, tree seed algorithm developed with tree seed algorithm was used for ten different test functions. The problem size was taken as 100 and the population numbers of the algorithm were accepted as 10, 30 and 50 respectively. The ST parameter value of the algorithm is accepted as its original value of 0.1. The algorithms were run in MATLAB program 30 times under the same conditions. When the obtained results were examined, it was seen that the improved tree seed algorithm obtained a better result than the original tree seed algorithm.

[1] Corresponding Author: Şırnak Üniversitesi, Müendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Şırnak, Türkiye, ORCID: 0000-0002-4842-3817, mehmetbes@sirnak.edu.tr

Keywords: Tree Seed Algorithm, Optimization, Test Functions.

1. Giriş

Bir optimizasyon algoritmanın güvenirliği ve verimliliği literatürde yer alan seçilmiş bir standart kriterler seti veya test fonksiyonları kullanılarak gerçekleştirilir (Jamil ve Yang, 2013). Her hangibir yeni optimizasyon için algoritmanın bir test fonksiyonları kümesi üzerinde uygulanması tavsiye edilmektedir (Gordon ve Whitley, 1993; Whitley vd., 1996). Bunula birlikte, bir algoritmanın problemler dizisi çok özelse başka algoritmalara karşı etkililiğinin çözdüğü problemlerle ölçülmeyeceği bir durum varsa bu işlem not edilmelidir (Jamil ve Yang, 2013). Bu nedenle, bir algoritmayı değerlendirmek için iyi performans gösterdiği problem türü tanımlanmalıdır. Özellikle büyük boyutlu problemlerde algoritmanın performans zayıflamaktadır (Weise vd., 2012). İlk önce karar değişkenlerinin sayısı arttıkça bir sorunun arama alanı katlanarak artar. İkinci olarak, arama alanın özellikleri boyut sayısı arttıkça değişebilmektedir (Li vd., 2013). Örneğin, Rosenbrock fonksiyonunu ele alırsak; bu problem iki boyutta tek biçimli bir işlevdir, ancak boyutsayısı arttıkça multimodal bir işleve dönüşmektedir (Hasenjäger vd., 2005). Ancak bazı araştırmacılar standart test fonksiyonlarının işlevlerinin çoğunun simetri ve ayrılabilirlik gibi doğal özelliklerinden dolayı nispeten kolay çözülebilir olması gerektiğini savunmaktadır (Whitley vd., 1996). Bu tarz yaklaşımlar olmasına rağmen günümüzde test fonksiyonları çeşitli alanlarda araştırmacıların sıklıkla kullandığı popüler fonksiyonlardır. Yeni geliştirilen meta-sezgisel algoritmalar test fonksiyonlarında mutlaka kullanılarak performans sıralaması yapılırlar. Bunun için son zamanlarda geliştirilen ağaç tohum algoritması (TSA) bu çalışma için kullanılarak performans alizleri gerçekleştirilmiş ve yakınsamam grafikleri elde edilmiştir.

2. Materyal and Metot

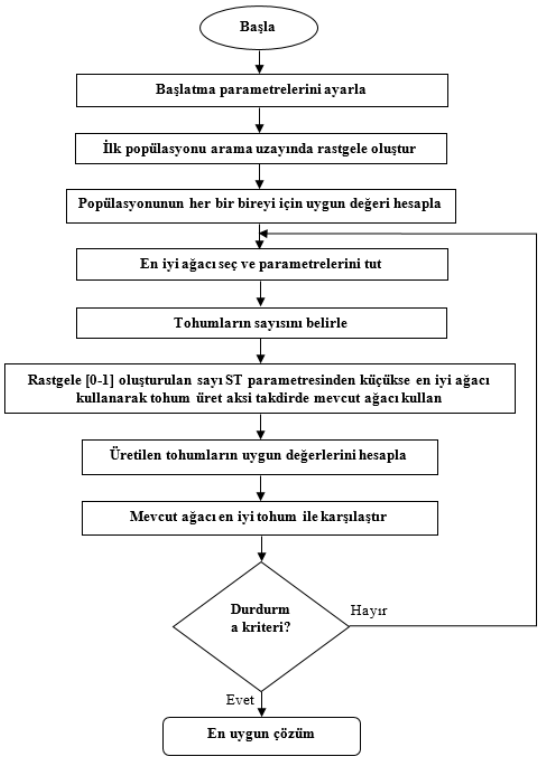
2.1. Ağaç Tohum Algoritması (TSA)

Ağaç tohum algoritması (TSA) 2015 yılında Kiran tarafından geliştirilmiş popülasyon temelli metasezgisel bir optimizasyon tekniğidir (Kiran, 2015). Bu algoritma sürekli optimizasyon problemlerinin çözümüne katkı sağlamak için geliştirilmiştir. TSA ağaç ve tohumlardan oluşmaktadır. Bunlar olası çözümleri ifade etmektedir. Algoritmanın başlangıç aşamasında ağaçlar arama uzayında rastgele oluşturulmaktadır (Kıran, 2016). Ağaç sayısı %10 ile %25 arasında olması tavsiye edilmektedir (Muneeswaran ve Rajasekaran, 2016). Tohumlar her ağaç için Denklem 1 ve Deklem 2 kullanılarak oluşturulmaktadır (Gungor vd., 2019).

$$S(k) = T(i) + \alpha(B - T(r)) \quad (1)$$

$$S(k) = T(i) + \alpha(T(i) - T(r)) \quad (2)$$

Burada S(k), T(i)'nin k. tohumudur. T(i) ise i. ağacı ifade etmektedir. α, -1 ile 1 arasında rastgele üretilen bir sayıdır. B, şu ana kadar elde edilen en iyi ağaçtır. T(r) ise T(i)'den farklı olan rastgele bir ağaç. Ayrıca algoritmada Arama Eğimi (ST) parametresi bulunmaktadır ve 0 ile 1 arasında değer alan bir parametredir ancak parameter değeri 0.1 kullanılmıştır. Arama işlemi sırasında 0 ile 1 arasında rastgele bir sayı üretilir ve ST ile karşılaştırılır. Karşılaştırma sonucunda ST bu rastgele sayıdan küçükse tohum oluşturmak için Denklem 1, eğer ST bu rastgele sayıdan büyükse tohum oluşturmak için Denklem 2 kullanılır. TSA'nın akış şeması Şekil 1'de verilmiştir (Beşkirli vd.).



Şekil 1. TSA'nın Akış Şeması

Beşkirli vd. tarafından 2019 yılında yapılan çalışmada tohum oluşumu esnasında ağaçların rastgele seçilmesi yerine turnuva seçim yöntemi kullanılmıştır. Bu çalışmada aynı yöntem kullanılmış ve 10 farklı test fonksiyonlarına uygulanmıştır. Bu yöntemde kullanılan formülasyon Denklem 3 ve Denklem 4'te verilmiştir (Beşkirli vd.).

$$S(k) = T(i) + \alpha(B - T(t)) \quad (3)$$

$$S(k) = T(i) + \alpha(T(i) - T(t)) \quad (4)$$

B, şu ana kadar elde edilen en iyi ağaçtır. T(t) ise T(i)'den farklı olan turnuva yöntemi ile elde edilmiş bir ağaçtır. Algoritmada Arama Eğimi (ST) parametre değeri 0.1 kullanılmıştır. Ancak arama işlemi sırasında 0 ile 1 arasında rastgele bir sayı üretilir ve ST ile karşılaştırılır. Karşılaştırma sonunda ST bu rastgele sayıdan küçükse tohum oluşturmak için Denklem 3, eğer ST bu rastgele sayıdan büyükse tohum oluşturmak için Denklem 4 kullanılır.

2.2. Test Fonksiyonları

Bu çalışmada on farklı test fonksiyonları kullanılmıştır. Bunlar Tablo 2'de fonksiyon isimleri ve arama uzayı aralığının değerleri sırasıyla verilmiştir.

Tablo 1. Test Fonksiyonları

Test Fonksiyonu	Fonksiyon Adı	Arama Uzayı
F1	Step	[-5.12, 5.12]
F2	Penalized 1	[-50, 50]
F3	Penalized 2	[-50, 50]
F4	Levy	[-10, 10]
F5	Weierstrass	[-0.5, 0.5]
F6	Schaffer	[-100, 100]
F7	Himmelblau	[-5, 5]
F8	Michalewicz	[0, π]
F9	Dixon-Price	[-10, 10]
F10	Schwefel 1.2	[-100, 100]

3. Bulgular

Bu çalışmada 10 farklı test fonksiyonu TSA ve TSA'nın geliştirilmiş hali olan G-TSA ile kullanılmıştır. On farklı fonksiyon 100 boyut için kullanılmıştır. Popülasyon büyüklükleri ise 10, 30 ve 50 olarak belirlenmiştir. Her iki algoritmanın on fonksiyon için elde

edilen en iyi, ortalama, en kötü ve standart sapma sonuçları Tablo 2, Tablo 3 ve Tablo 4'te verilmiştir. Her popülasyon için algoritmaların fonksiyonlara göre yakınsama grafikleri sırasıyla Şekil 2'de verilmiştir.

Tablo 2. Popülasyon sayısı 10 Alınarak Elde Edilen Sonuçlar

Fonksiyon	Değer	TSA	G-TSA
F1	En İyi	**0.00E+00**	1.00E+00
	Ortalama	1.80E+00	5.50E+00
	En Kötü	5.00E+00	1.20E+01
	Standart Sapma	1.54E+00	3.77E+00
F2	En İyi	1.93E-09	**6.22E-28**
	Ortalama	1.21E+00	5.92E-02
	En Kötü	5.91E+00	4.05E-01
	Standart Sapma	2.00E+00	1.24E-01
F3	En İyi	**8.04E+00**	3.29E+01
	Ortalama	2.97E+01	4.37E+01
	En Kötü	5.74E+01	7.71E+01
	Standart Sapma	1.34E+01	1.26E+01
F4	En İyi	1.20E-21	**2.66E-29**
	Ortalama	2.20E-02	2.20E-02
	En Kötü	1.10E-01	1.10E-01
	Standart Sapma	4.39E-02	4.39E-02
F5	En İyi	**2.84E-14**	1.16E-01
	Ortalama	5.50E-01	2.71E+00
	En Kötü	6.00E+00	1.08E+01
	Standart Sapma	1.10E+00	3.13E+00
F6	En İyi	2.28E-01	**1.78E-01**
	Ortalama	2.50E-01	2.30E-01
	En Kötü	3.12E-01	4.52E-01
	Standart Sapma	2.90E-02	7.74E-02
F7	En İyi	**-7.30E+01**	-7.10E+01
	Ortalama	-7.04E+01	-6.94E+01
	En Kötü	-6.76E+01	-6.76E+01
	Standart Sapma	1.96E+00	1.20E+00
F8	En İyi	1.47E-17	**7.00E-18**
	Ortalama	1.76E-16	8.88E-17
	En Kötü	7.44E-16	1.87E-16
	Standart Sapma	2.26E-16	5.82E-17
F9	En İyi	**6.67E-01**	**6.67E-01**
	Ortalama	1.01E+01	1.12E+01
	En Kötü	2.79E+01	2.01E+01
	Standart Sapma	8.25E+00	6.19E+00
F10	En İyi	7.87E+04	**1.20E+04**
	Ortalama	1.92E+05	6.29E+04
	En Kötü	2.69E+05	2.14E+05
	Standart Sapma	5.33E+04	7.58E+04

Tablo 3. Popülasyon sayısı 30 Alınarak Elde Edilen Sonuçlar

Fonksiyon	Değer	TSA	G-TSA
F1	En İyi	1.87E+02	**0.00E+00**
	Ortalama	2.95E+02	1.00E-01
	En Kötü	4.82E+02	1.00E+00
	Standart Sapma	9.95E+01	3.00E-01

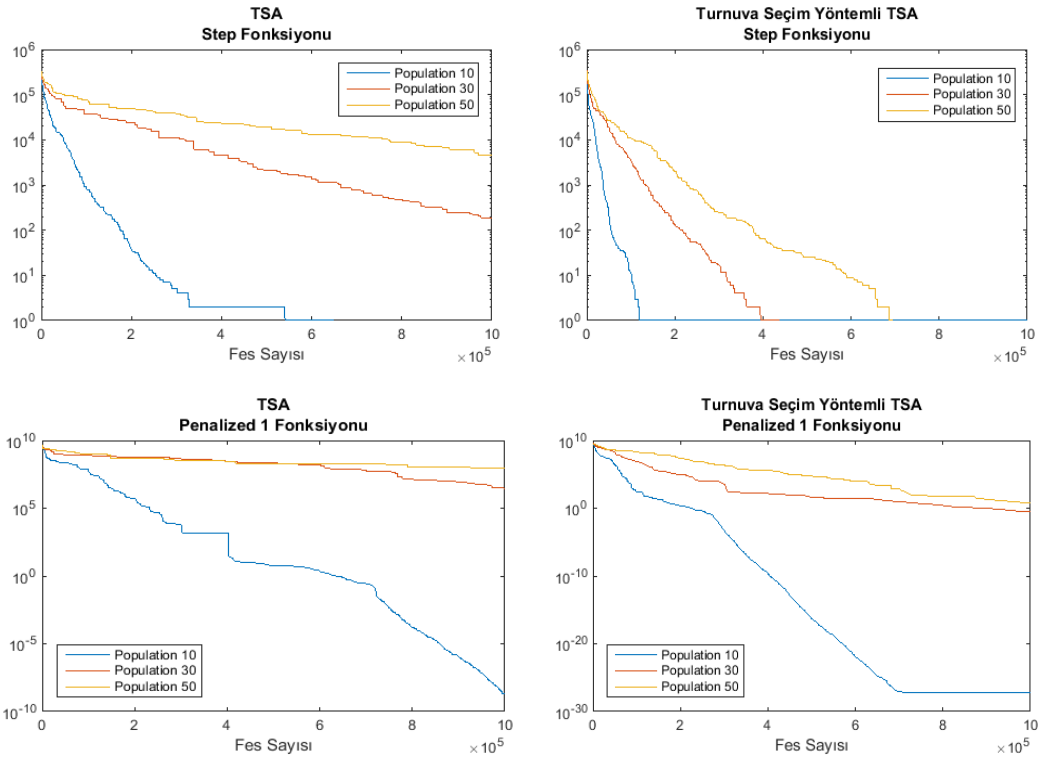
F2	En İyi	2.76E+06	**2.96E-01**
	Ortalama	2.97E+07	1.26E+00
	En Kötü	6.92E+07	5.23E+00
	Standart Sapma	2.38E+07	1.39E+00
F3	En İyi	5.76E+06	**1.87E+01**
	Ortalama	4.94E+07	4.69E+01
	En Kötü	1.12E+08	7.38E+01
	Standart Sapma	4.12E+07	1.70E+01
F4	En İyi	7.94E+00	**8.64E-16**
	Ortalama	1.70E+01	4.39E-02
	En Kötü	2.40E+01	1.10E-01
	Standart Sapma	4.83E+00	5.38E-02
F5	En İyi	6.89E+00	**2.09E-11**
	Ortalama	9.07E+00	7.21E-10
	En Kötü	1.30E+01	2.96E-09
	Standart Sapma	1.82E+00	8.74E-10
F6	En İyi	4.99E-01	**2.28E-01**
	Ortalama	4.99E-01	2.71E-01
	En Kötü	5.00E-01	3.12E-01
	Standart Sapma	2.21E-04	3.21E-02
F7	En İyi	-7.05E+01	**-7.30E+01**
	Ortalama	-6.91E+01	-7.07E+01
	En Kötü	-6.81E+01	-6.96E+01
	Standart Sapma	6.95E-01	1.00E+00
F8	En İyi	1.03E-10	**2.53E-18**
	Ortalama	1.05E-08	1.44E-17
	En Kötü	5.13E-08	3.08E-17
	Standart Sapma	1.53E-08	8.33E-18
F9	En İyi	4.73E+04	**7.42E-01**
	Ortalama	1.04E+05	1.16E+01
	En Kötü	3.21E+05	2.47E+01
	Standart Sapma	7.76E+04	7.85E+00
F10	En İyi	2.65E+05	**1.22E+05**
	Ortalama	3.04E+05	1.81E+05
	En Kötü	3.41E+05	2.44E+05
	Standart Sapma	2.17E+04	4.34E+04

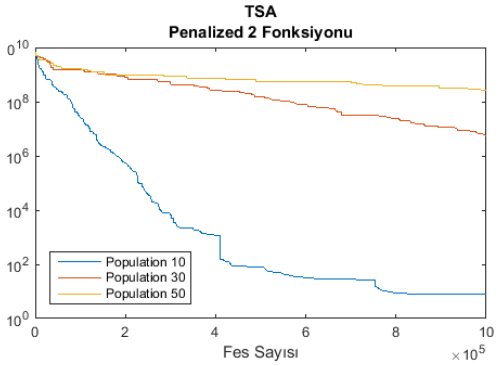
Tablo 4. Popülasyon sayısı 50 Alınarak Elde Edilen Sonuçlar

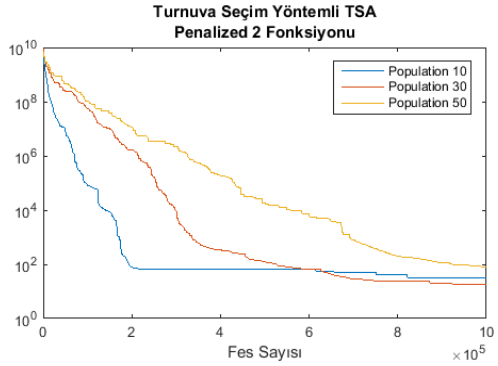
Fonksiyon	Değer	TSA	G-TSA
F1	En İyi	4.51E+03	**0.00E+00**
	Ortalama	5.04E+03	0.00E+00
	En Kötü	6.05E+03	0.00E+00
	Standart Sapma	4.79E+02	0.00E+00
F2	En İyi	8.44E+07	**6.18E+00**
	Ortalama	1.92E+08	3.64E+01
	En Kötü	2.83E+08	1.09E+02
	Standart Sapma	7.17E+07	2.74E+01
F3	En İyi	2.53E+08	**7.93E+01**
	Ortalama	4.11E+08	1.70E+02
	En Kötü	5.50E+08	3.08E+02
	Standart Sapma	7.44E+07	8.41E+01
F4	En İyi	1.04E+02	**6.76E-08**

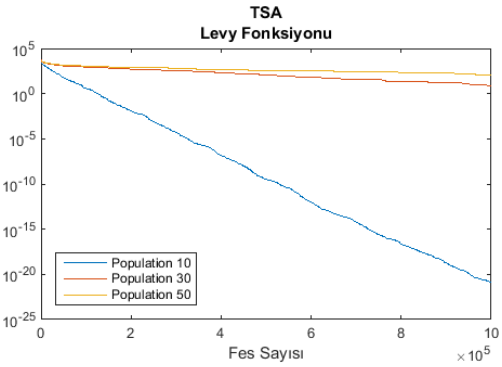
	Ortalama	1.65E+02	1.10E-02
	En Kötü	2.25E+02	1.10E-01
	Standart Sapma	3.66E+01	3.30E-02
F5	En İyi	3.92E+01	**1.23E-03**
	Ortalama	5.30E+01	2.23E-03
	En Kötü	6.48E+01	4.40E-03
	Standart Sapma	6.94E+00	8.13E-04
F6	En İyi	5.00E-01	**3.46E-01**
	Ortalama	5.00E-01	4.00E-01
	En Kötü	5.00E-01	4.43E-01
	Standart Sapma	5.35E-05	3.29E-02
F7	En İyi	-5.84E+01	**-7.27E+01**
	Ortalama	-5.29E+01	-7.09E+01
	En Kötü	-4.28E+01	-6.84E+01
	Standart Sapma	4.58E+00	1.25E+00
F8	En İyi	8.49E-07	**3.39E-18**
	Ortalama	9.45E-06	1.87E-17
	En Kötü	3.57E-05	4.90E-17
	Standart Sapma	1.16E-05	1.35E-17
F9	En İyi	7.51E+05	**6.29E+00**
	Ortalama	1.11E+06	1.44E+01
	En Kötü	1.59E+06	2.41E+01
	Standart Sapma	2.58E+05	6.79E+00
F10	En İyi	2.76E+05	**1.44E+05**
	Ortalama	3.15E+05	2.21E+05
	En Kötü	3.60E+05	2.57E+05
	Standart Sapma	2.00E+04	4.04E+04

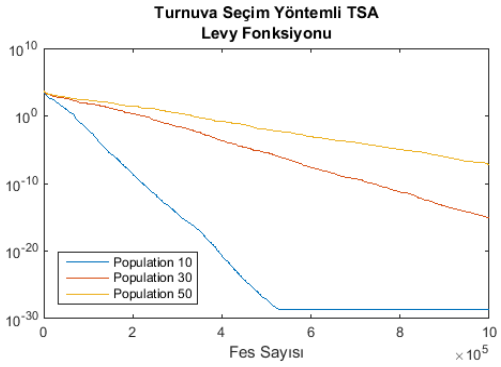
Tablolar incelendiğinde popülasyon sayısı 10 olduğu durumda fonksiyonların yarısının G-TSA tarafından daha iyi bir çözüm elde ettiği Tablo 2'de görülmüştür. Popülasyon sayısı 30 ve 50 olduğu durumda ise G-TSA bütün fonksiyonlar için en iyi çözümü elde etmiştir. Böylece popülasyon sayısı arttıkça G-TSA'nın performansının da arttığı görülmüştür.

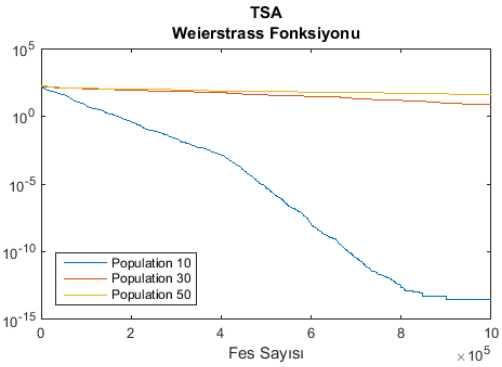


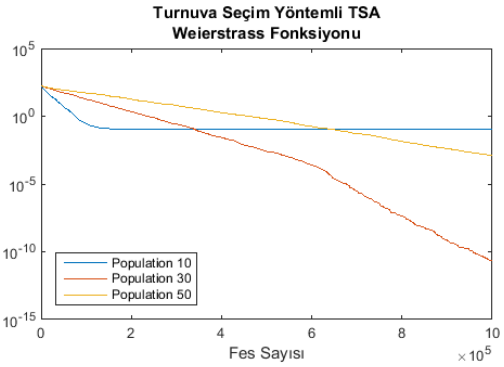

TSA
Penalized 2 Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5

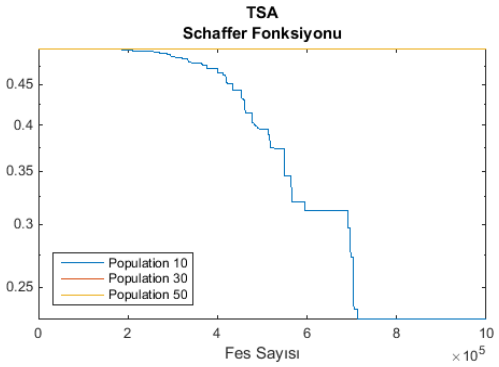

Turnuva Seçim Yöntemli TSA
Penalized 2 Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5

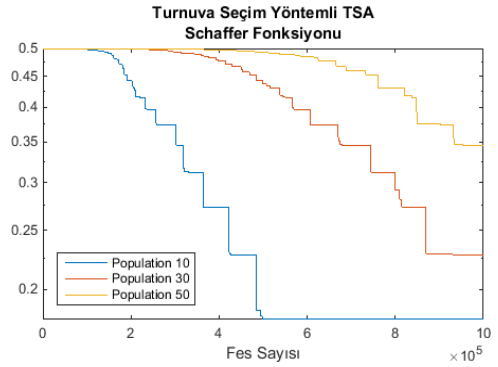

TSA
Levy Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5

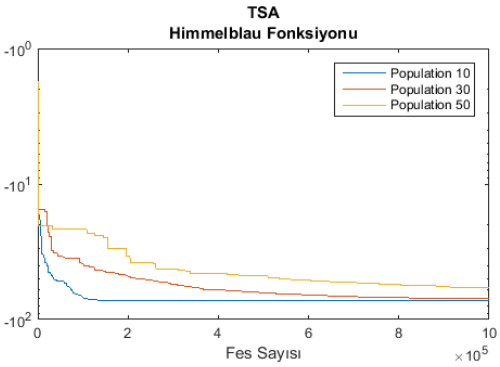

Turnuva Seçim Yöntemli TSA
Levy Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5

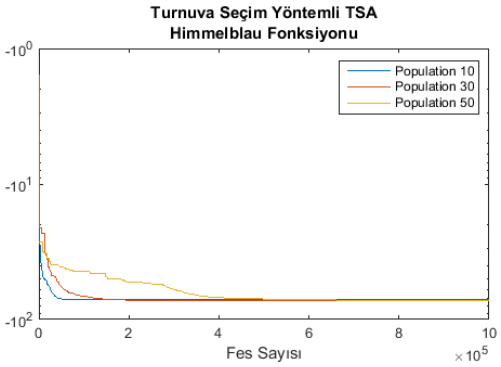

TSA
Weierstrass Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5

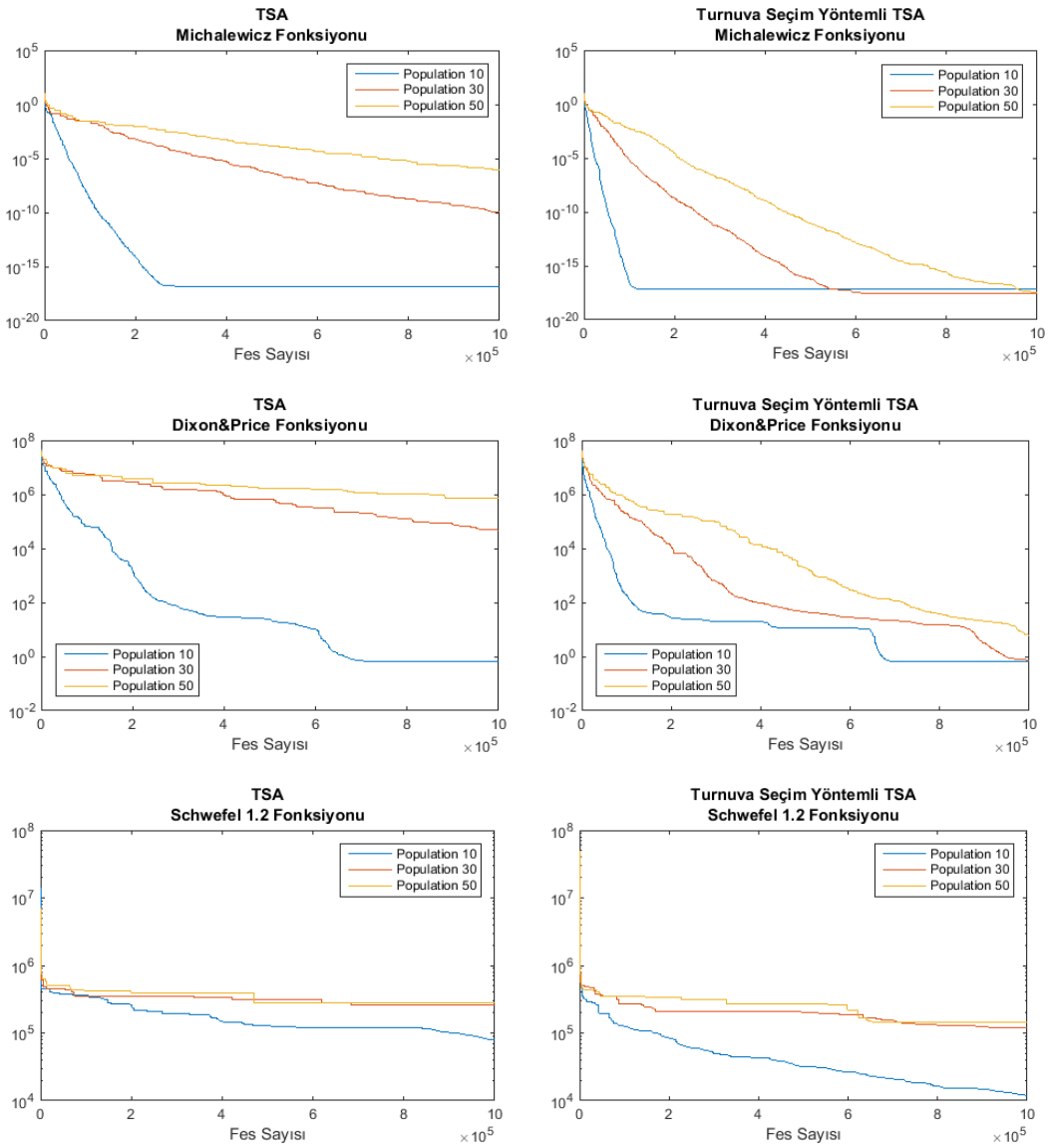

Turnuva Seçim Yöntemli TSA
Weierstrass Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5


TSA
Schaffer Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5


Turnuva Seçim Yöntemli TSA
Schaffer Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5


TSA
Himmelblau Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5


Turnuva Seçim Yöntemli TSA
Himmelblau Fonksiyonu
Population 10
Population 30
Population 50
Fes Sayısı
×10^5



Şekil 2. Fonksiyonların Yakınsama Grafikleri

Şekil 2'de yakınsama grafikleri incelendiğinde popülasyon sayısı düştükçe her iki algoritma için hızlı bir yakınsama gerçekleştiği görülmektedir. Ancak sonuçlara bakıldığında popülasyon sayısının artması G-TSA'nın daha iyi sonuç elde etmesini sağlamıştır.

4. Sonuçlar

Geliştirilmiş TSA ve orijinal TSA algoritmaları on farklı test fonksiyonları üzerinde performans analizi gerçekleştirilmiştir. 10, 30 ve 50 popülasyon sayısı ile 100 boyut kullanılarak yapılan bu çalışma 30 kez aynı şartlarda çalıştırılarak gerçekleştirilmiştir. G-TSA'nın TSA'ya göre başarısı tablolarda gösterilerek yakınsama grafikleri eklenmiştir. Buradan ortaya çıkan sonuç algoritmanın popülasyon sayısı arttıkça G-TSA'nın elde ettiği performansın TSA'ya göre daha iyi olduğu görülmüştür. Bundan dolayı G-TSA'nın sürekli problemler için uygulanabilir bir algoritma olduğu varsayılabilir. G-TSA, çok büyük boyutlu problemlere uygulanarak TSA'ya göre başarısı karşılaştırılabilir ve böylece G-TSA'nın çok boyutlu sürekli optimizasyon problemleri için varsayılan uygunluğu araştırılabilir.

5. Açıklama

Bu çalışma Şırnak Üniversitesi Bilimsel Araştırma Projeleri Koordinasyon Birimi tarafından 2019.FNAP.06.01.01 numaralı araştırma projesi ile desteklenmiştir.

Kaynaklar

Beşkirli, A., Özdemir, D., & Temurtaş, H. A comparison of modified tree–seed algorithm for high-dimensional numerical functions. *Neural Computing and Applications*, 1-35.

Gordon, V. S., & Whitley, D. (1993). *Serial and parallel genetic algorithms as function optimizers.* Paper presented at the ICGA.

Gungor, I., Emiroglu, B. G., Cinar, A. C., & Kiran, M. S. (2019). Integration search strategies in tree seed algorithm for high dimensional function optimization. *International Journal of Machine Learning and Cybernetics*, 1-19.

Hasenjäger, M., Sendhoff, B., Sonoda, T., & Arima, T. (2005). *Three dimensional evolutionary aerodynamic design optimization with CMA-ES.* Paper presented at the Proceedings of the 7th annual conference on Genetic and evolutionary computation.

Jamil, M., & Yang, X.-S. (2013). A literature survey of benchmark functions for global optimization problems. *arXiv preprint arXiv:1308.4008*.

Kiran, M. S. (2015). TSA: Tree-seed algorithm for continuous optimization. *Expert Systems with Applications, 42*(19), 6686-6698.

Kıran, M. S. (2016). An implementation of tree-seed algorithm (TSA) for constrained optimization *Intelligent and Evolutionary Systems* (pp. 189-197): Springer.

Li, X., Tang, K., Omidvar, M. N., Yang, Z., Qin, K., & China, H. (2013). Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization. *gene, 7*(33), 8.

Muneeswaran, V., & Rajasekaran, M. P. (2016). *Performance evaluation of radial basis function networks based on tree seed algorithm.* Paper presented at the 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT).

Weise, T., Chiong, R., & Tang, K. (2012). Evolutionary optimization: Pitfalls and booby traps. *Journal of Computer Science and Technology, 27*(5), 907-936.

Whitley, D., Rana, S., Dzubera, J., & Mathias, K. E. (1996). Evaluating evolutionary algorithms. *Artificial intelligence, 85*(1-2), 245-276.